WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Friday, February 03, 2006

Hide?	Set Name	Query	Hit Count
	DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ		
☐	L16	717/101.ccls. and (object near (model or data)).ab.	4
☐	L15	717/123.ccls. and (object near (model or data)).ab.	1
☐	L14	717/104.ccls. and (object near (model or data)).ab.	27
☐	L13	L9 and 20040205552	1
☐	L12	L8 and 20040205552	2
☐	L11	L9 and bidirection$	3
☐	L10	L9 and bidirectional$	3
☐	L9	L8 and (one-to-many or one?to?many or "one to many")	18
☐	L8	L4 and (object near (model or data))	209
	DB=EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ		
☐	L7	L4 and (object near (model or data))	12
☐	L6	L5 and (object near (model or data))	12
☐	L5	L4	152
	DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ		
☐	L4	(map$ or transfor$ or conver$ or translat$) near3 (markup language)	747
☐	L3	717/120-123.ccls. and ((map$ or transfor$ or conver$ or translat$) near3 (markup language))	3
☐	L2	717/104-109.ccls. and ((map$ or transfor$ or conver$ or translat$) near3 (markup language))	19
☐	L1	717/101-102.ccls. and ((map$ or transfor$ or conver$ or translat$) near3 (markup language))	5

END OF SEARCH HISTORY

WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Friday, February 03, 2006

Hide?	Set Name	Query	Hit Count
		DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ	
☐	L8	L7 and (request$ and respon$).clm. attach claimed search. ←	2
☐	L7	L6 and key.clm.	7
☐	L6	L5 and (metadata or meta?data).clm.	32
☐	L5	L4 or L3 or L2 or L1	453
☐	L4	((map$ or conver$ or translat$ or transform$) near (object model)).clm.	39
☐	L3	((map$ or conver$ or translat$ or transform$) near (XML)).clm.	272
☐	L2	((map$ or conver$ or translat$ or transform$) near (HTML)).clm.	89
☐	L1	((map$ or conver$ or translat$ or transform$) near (markup language)).clm.	61

END OF SEARCH HISTORY

suitable manner in one or more embodiments. In the following description, numerous specific details are provided, such as examples of programming, software components, user selections, network transactions, database queries, database structures, hardware components, hardware circuits, hardware chips, etc., to provide a thorough understanding of embodiments of the invention. One skilled in the relevant art will recognize, however, that the invention may be practiced without one or more of the specific details, or with other methods, components, materials, and so forth. In other instances, well-known structures, materials, or operations are not shown or described in detail to avoid obscuring aspects of the invention.

[0108] The schematic flow chart diagrams included are generally set forth as logical flow chart diagrams. As such, the depicted order and labeled steps are indicative of one embodiment of the presented method. Other steps and methods may be conceived that are equivalent in function, logic, or effect to one or more steps, or portions thereof, of the illustrated method. Additionally, the format and symbols employed are provided to explain the logical steps of the method and are understood not to limit the scope of the method. Although various arrow types and line types may be employed in the flow chart diagrams, they are understood not to limit the scope of the corresponding method. Indeed, some arrows or other connectors may be used to indicate only the logical flow of the method. For instance, an arrow may indicate a waiting or monitoring period of unspecified duration between enumerated steps of the depicted method. Additionally, the order in which a particular method occurs may or may not strictly adhere to the order of the corresponding steps shown.

[0109] The present invention may be embodied in other specific forms without departing from its spirit or essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive. The scope of the invention is, therefore, indicated by the appended claims rather than by the foregoing description. All changes which come within the meaning and range of equivalency of the claims are to be embraced within their scope.

What is claimed is:

1. A programmed method for facilitating transactions between thin-clients and Message Format Service (MFS)-based Information Management System (IMS) applications, the programmed method comprising the process steps of:

storing conversation attributes associated with a conversational transaction between a thin-client and an MFS-based IMS application, the conversation attributes comprising connection information and conversation-specific information;

preprocessing one or more transaction messages from the thin-client based on a transaction message type;

updating the stored conversation attributes in response to a change in the conversation attributes caused by the one or more transaction messages; and

formatting a conversation output message for the thin-client.

2. The programmed method of claim 1 wherein said programmed method is in the form of process steps.

3. The programmed method of claim 1 wherein said programmed method is in the form of a computer-readable medium embodying computer instructions for performing the process steps.

4. The programmed method of claim 1 wherein said programmed method is in the form of a computer system programmed by software, hardware, firmware, or any combination thereof, for performing the process steps.

5. The programmed method of claim 1 wherein said programmed method is in the form of an apparatus comprising software, hardware, firmware, or any combination thereof, for performing the process steps.

6. The programmed method of claim 1 further comprising selectively transmitting conversation input messages to a MFS-based IMS applications based on the transaction message type and conversation attributes.

7. The programmed method of claim 6 wherein conversation input messages are transmitted through an MFS adapter configured to mediate conversational transactions and conversation output messages are received from the MFS adapter.

8. The programmed method of claim 1 wherein formatting a conversation output message comprises generating HTML data from thin-client specific XML Metadata Interchange (XMI) information provided by a conversational transaction mediator and XML Stylesheet Language (XSL) information and the programmed method further comprising sending the HTML data to the thin-client.

9. The programmed method of claim 8 wherein the XMI information is associated with a particular MFS-based IMS application and the XSL information is user configurable.

10. The programmed method of claim 1 wherein the one or more transaction messages comprise a conversation command and preprocessing comprises selectively modifying connection information in response to the conversation command.

11. The programmed method of claim 1 wherein the one or more transaction messages comprise a paging command and preprocessing comprises selectively modifying the conversation output message based on pagination information associated with the conversation-specific information.

12. A system for facilitating conversational transactions between thin-clients and Message Format Service (MFS)-based Information Management System (IMS) applications, comprising:

an IMS interface configured to execute on a mainframe operating system and enable conversational transactions between an MFS-based IMS application and a Transmission Control Protocol/Internet Protocol (TCP/IP) client;

a web module configured to operate on a web application server as the TCP/IP client and translate between XML conversation messages and a byte stream compatible with the IMS interface;

a conversational transaction servlet configured to operate on a web application server, the conversational transaction servlet further comprising,

a state module configured to store conversation attributes associated with conversational transactions between a thin-client and the MFS-based IMS application over a network, and update the stored

conversation attributes in response to a change in the conversation attributes caused by one or more transaction messages;

a preprocessor configured to preprocess one or more transaction messages from the thin-client based on a transaction message type; and

a formatter configure to format a conversation output message for the thin-client according to the conversation attributes.

13. The system of claim 12 wherein the formatter combines XMI information from the web module with XSL information to generate a HTML data suitable for display on the thin-client.

14. The system of claim 12 wherein the one or more transaction messages comprise a conversation command selected from the group consisting of "EXIT,""HOLD," and "RELEASE" and the preprocessor selectively modifies connection information in response to the conversation command.

15. The system of claim 12 wherein the one or more transaction messages comprise a paging command directed to one of a logical page and a physical page, the preprocessor selectively modifying the conversation output message based on pagination information associated with the conversation-specific information.

16. An apparatus for facilitating transactions between thin-clients and Message Format Service (MFS)-based Information Management System (IMS) applications, comprising:

a security module configured to preserve and maintain user security credentials;

a connection module configured to establish a connection with an MFS-based IMS application by way of an MFS adapter, the connection supporting a conversational transaction;

a state module configured to preserve and maintain connection information associated with the connection and conversation-specific information based on one or more transaction messages from the thin-client; and

a control module configured to process a transaction message having one or more transaction message types.

17. The apparatus of claim 16, wherein the control module further comprises,

a function key module configured to process a transaction message having a function key indicator;

a page module configured to process a transaction message having a paging request;

a command module configured to process a transaction message having a conversational command; and

a formatter configure to format a conversation output message for the thin-client based on presentation information provided by the MFS-based IMS application.

18. The apparatus of claim 17, wherein the formatter determines a subset of XMI information from an XMI repository based on page information within the conversation-specific information and combines the subset of XMI with XSL information to produce HTML suitable for presentation by the thin-client.

19. The apparatus of claim 16, wherein the security module authenticates the security credentials with a mainframe security control subsystem.

20. The apparatus of claim 16, wherein the connection module manages a plurality of connections in response to conversational commands.

* * * * *

L5: Entry 3 of 5 File: PGPB Feb 6, 2003

PGPUB-DOCUMENT-NUMBER: 20030028551
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20030028551 A1

TITLE: System and method for retrieval of objects from object to relational mappings

PUBLICATION-DATE: February 6, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY
Sutherland, James Bryce	Ottawa		CA

US-CL-CURRENT: 707/200

CLAIMS:

What is claimed is:

1. A method for retrieving target objects stored in a relational database to which an object model is mapped, the method comprising steps of: generating a retrieval query to read target objects for a collection of source objects, the collection of source objects having many-to-many relationships with the target objects, the collection of source objects and target objects being respectively stored in one or more source tables and target tables in the database, and the many-to-many relationship being defined in the database by using an intermediate join table of the source tables and the target tables; selecting join table information from the many-to-many join table relating to the collection of source objects and the target objects to enable matching of the target objects and the source objects using the join table information; and retrieving the matched target objects by executing the retrieval query on the database.

2. The method as claimed in claim 1 further comprising steps of: specifying batch readable relationships on a source query for reading the collection of source objects; generating a nested query for reading related objects nested in the target objects; appending query information of the target objects to the nested query; and retrieving the related objects by executing the nested query.

3. The method as claimed in claim 1 wherein the generating step comprises steps of: obtaining a source expression tree relating to the collection of the source objects; building a target expression tree defined by the many-to-many mapping including a join between the target tables and the join table; combining the source expression tree and the target expression tree to produce a combined expression tree; and generating the retrieval query based on the combined expression tree.

4. The method as claimed in claim 3 wherein the target expression tree building step obtains the target expression tree from mapping meta-data which contains information as to how object classes and relationships of the object model map to tables and foreign keys in the database.

5. The method as claimed in claim 4 wherein the target expression tree building step obtains the target expression tree from mapping meta-data which includes a list of key and value pairs

of the many-to-may join table.

6. The method as claimed in claim 1 wherein the selecting step comprises steps of: executing the retrieval query on the database for reading the target objects; obtaining target object information and join table information from the join table; and appending the target object information and the join table information to the retrieval query.

7. The method as claimed in claim 6 wherein the join table information including foreign key values and the appending step appends the foreign key values to the retrieval query.

8. The method as claimed in claim 6 wherein the appending step appends the target table information and the join table information to a select clause of a select statement.

9. The method as claimed in claim 6 wherein the retrieving step comprises steps of: obtaining the target objects; and populating relationships of the source objects with the target objects by comparing a primary key value of each source object with a foreign key value of each target object using the foreign key values stored in the retrieval query; and matching each source object with matched target objects.

10. A method for retrieving objects stored in a relational database to which an object model is mapped, the method comprising steps of: obtaining nested specification information representing joins relating to a source object and related objects which are joined with the source object with multi-level relationships; obtaining parent query information representing a parent query for reading one or more parent objects at a parent level; generating a nested query for querying objects of next lower level which is next lower than the parent level; appending to the nested query the parent query information and the joins using the nested specification information; and retrieving the objects of next lower level by executing the nested query on the database.

11. The method as claimed in claim 10, wherein the nested specification obtaining step obtains the nested specification information from mapping meta-data which contains information as to how object classes and relationships of the object model map to tables and foreign keys in the database.

12. The method as claimed in claim 10 further comprising a step of specifying batch readable relationships to the parent query for allowing batch reading of the related objects.

13. The method as claimed in claim 12, wherein the specifying step comprises a step of determining the batch readable relationships based on the nested specification.

14. The method as claimed in claim 10 further comprising a step of setting automatic batch reading for automatically generating the nested query for reading objects of lower levels.

15. The method as claimed in claim 10, wherein the generating step generates a single query for each type of relationships at each level.

16. The method as claimed in claim 10, wherein the retrieving step further comprising a step of delaying execution of the nested query until the relationship of the source object is accessed.

17. A retrieval system for retrieving target objects stored in a relational database to which an object model is mapped, the retrieval system comprising: a query generator for generating a retrieval query to read target objects for a collection of source objects, the collection of source objects having many-to-many relationships with the target objects, the collection of source objects and target objects being respectively stored in one or more source tables and target tables in the database, and the many-to-many relationship being defined in the database by using an intermediate join table of the source tables and the target tables; a join table information handler for selecting join table information from the many-to-many join table

relating to the collection of source objects and the target objects to enable matching of the target objects and the source objects using the join table information; and a batch reading handler for retrieving the matched target objects by executing the retrieval query on the database.

18. The retrieval system as claimed in claim 17, wherein the query generator comprises: an expression tree handler for obtaining a source expression tree relating to the collection of the source objects, and a target expression tree defined by the many-to-many mapping including a join between the target tables and the join table; an expression tree combiner for combining the source expression tree and the target expression tree to produce a combined expression tree for generating the retrieval query based on the combined expression tree.

19. The retrieving system as claimed in claim 17, wherein the expression tree handler obtains mapping meta-data which contains information as to how object classes and relationships of the object model map to tables and foreign keys in the database.

20. The retrieval system as claimed in claim 17, wherein the join table information handler obtains target object information and join table information from the join table; and the batch reading handler appends to the retrieval query target object information and the join table information.

21. The retrieval system as claimed in claim 20, wherein the join table information handler obtains foreign key values.

22. The retrieval system as claimed in claim 21, wherein the batch reading handler appends the foreign key values to the retrieval query.

23. The retrieval system as claimed in claim 22, wherein the batch reading handler has a comparator for comparing a primary key value of each source object with a foreign key value of each target object using the foreign key values appended to the retrieval query; and matching each source object with matched target objects.

24. A retrieving system for retrieving objects stored in a relational database to which an object model is mapped, the retrieval system comprising: an information receiver for obtaining nested specification information representing joins relating to the source object and related objects which are joined with the source object with multi-level relationships; a query generator for generating a nested query for querying objects of next lower level to parent objects which are queried by a parent query; and a batch reading handler for appending to the nested query information of the parent query and the joins using the nested specification information, and retrieving the objects of next lower level by executing the nested query on the database.

25. The retrieval system as claimed in claim 24, wherein the information receiver obtains the nested specification information from mapping meta-data which contains information as to how object classes and relationships of the object model map to tables and foreign keys in the database.

26. The retrieval system as claimed in claim 24, wherein the batch reading handler has a batch reading setter for allowing batch reading of the related objects.

27. The retrieval system as claimed in claim 26, wherein the batch reading setter specifies batch readable relationships to the parent query for allowing batch reading.

28. The retrieval system as claimed in claim 27, wherein the batch reading setter determines the batch readable relationships based on the nested specification.

29. The retrieval system as claimed in claim 26, wherein the batch reading setter sets

automatic batch reading for automatically generating the nested query for reading objects of lower levels.

30. The retrieval system as claimed in claim 24, wherein the batch reading handler has an indirection function setter for delaying execution of the nested query until the relationship of the source object is accessed.

31. Computer media storing the instructions or statements for use in the execution in a computer of a method method for retrieving target objects stored in a relational database to which an object model is mapped, the method comprising steps of: generating a retrieval query to read target objects for a collection of source objects, the collection of source objects having many-to-many relationships with the target objects, the collection of source objects and target objects being respectively stored in one or more source tables and target tables in the database, and the many-to-many relationship being defined in the database by using an intermediate join table of the source tables and the target tables; selecting join table information from the many-to-many join table relating to the collection of source objects and the target objects to enable matching of the target objects and the source objects using the join table information; retrieving the matched target objects by executing the retrieval query on the database.

32. Electronic signals for use in the execution in a computer of a method for retrieving target objects stored in a relational database to which an object model is mapped, the method comprising steps of: generating a retrieval query to read target objects for a collection of source objects, the collection of source objects having many-to-many relationships with the target objects, the collection of source objects and target objects being respectively stored in one or more source tables and target tables in the database, and the many-to-many relationship being defined in the database by using an intermediate join table of the source tables and the target tables; selecting join table information from the many-to-many join table relating to the collection of source objects and the target objects to enable matching of the target objects and the source objects using the join table information; retrieving the matched target objects by executing the retrieval query on the database.

33. A computer program product for use in the execution in a computer of method for retrieving target objects stored in a relational database to which an object model is mapped, the product comprising: a module for generating a retrieval query to read target objects for a collection of source objects, the collection of source objects having many-to-many relationships with the target objects, the collection of source objects and target objects being respectively stored in one or more source tables and target tables in the database, and the many-to-many relationship being defined in the database by using an intermediate join table of the source tables and the target tables; a module for selecting join table information from the many-to-many join table relating to the collection of source objects and the target objects to enable matching of the target objects and the source objects using the join table information; a module for retrieving the matched target objects by executing the retrieval query on the database.

34. Computer media storing the instructions or statements for use in the execution in a computer of a method for retrieving objects stored in a relational database to which an object model is mapped, the method comprising steps of: obtaining nested specification information representing joins relating to the source object and related objects which are joined with the source object with multi-level relationships; obtaining parent query information representing a parent query for reading one or more parent objects at a parent level; generating a nested query for querying objects of next lower level which is next lower than the parent level; appending to the nested query the parent query information and the joins using the nested specification information; and retrieving the objects of next lower level by executing the nested query on the database.

35. Electronic signals for use in the execution in a computer of a method for retrieving objects stored in a relational database to which an object model is mapped, the method comprising steps of: obtaining nested specification information representing joins relating to

the source object and related objects which are joined with the source object with multi-level relationships; obtaining parent query information representing a parent query for reading one or more parent objects at a parent level; generating a nested query for querying objects of next lower level which is next lower than the parent level; appending to the nested query the parent query information and the joins using the nested specification information; and retrieving the objects of next lower level by executing the nested query on the database.

36. A computer program product for use in the execution in a computer of method for retrieving objects stored in a relational database to which an object model is mapped, the product comprising: a module for obtaining nested specification information representing joins relating to the source object and related objects which are joined with the source object with multi-level relationships; a module for obtaining parent query information representing a parent query for reading one or more parent objects at a parent level; a module for generating a nested query for querying objects of next lower level which is next lower than the parent level; a module for appending to the nested query the parent query information and the joins using the nested specification information; and a module for retrieving the objects of next lower level by executing the nested query on the database.

Previous Doc Next Doc Go to Doc#

L5: Entry 1 of 5 File: PGPB Jul 7, 2005

PGPUB-DOCUMENT-NUMBER: 20050149555
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20050149555 A1

TITLE: System and method for managing object to relational one-to-many mapping

PUBLICATION-DATE: July 7, 2005

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY
Wang, Yaoping	Ottawa		CA
Sutherland, James Bryce	Ottawa		CA

US-CL-CURRENT: 707/103R

CLAIMS:

1-24. (canceled)

25. A method for managing object to relational one-to-many mapping for an object model mapped to a relational database, the method comprising a step of tracking changes to relationships and target objects with which the source object has one-to-many relationships of privately owned type, wherein the tracking step comprises steps of: creating, at the start of a transaction, a source clone of a source object having a primary key value, a relationship clone of a relationship in the source object, and a target clone of target objects referenced by the relationship; determining, when the transaction is committed, changes to the source object and target objects by comparing the source clone to the current state of the source object, and comparing the target clone to the current state of the target objects using the primary key value of the source object added to the target objects, and updating the relationship between the source object and the target objects by: determining which objects have been added, removed or changed in the relationship by examining mapping meta-data including information of corresponding source table, one or more target tables and foreign keys of the target tables, generating an INSERT instruction for each target object of a source target that has been added, generating a DELETE instruction for each target object of a source target that has been removed, and generating an UPDATE instruction for each target object of a source target that has been changed.

26. (canceled)

27. The method as claimed in claim 25, wherein the steps of generating an insert, delete and update instructions comprise steps of: generating SQL Insert statements for objects that have been added; generating SQL Delete statements for objects that have been removed; and generating SQL Update statements for objects that have been changed.

28. The method as claimed in claim 25 further comprising a step of: storing the mapping meta-data external to the source object class and the target object classes.

29. The method as claimed in claim 28, wherein the storing step stores the mapping meta-data as

XML files.

30. A one-to-many mapping manager for managing object to relational one-to-many mapping for an object model mapped to a relational database, the system comprising: a meta-data receiver for obtaining, for a source data having a primary key value and being manipulated, mapping meta-data including information of one or more corresponding target tables for storing multiple target objects with which the source object has one-to-many relationships of privately owned type and information of one or more foreign keys of the corresponding target tables; an instruction generator for generating instructions to manage the multiple target objects and the relationship based on the mapping meta-data, wherein the instruction generator is configured to generate update instructions to track changes to the multiple target objects and the relationship, and to update the database when a source object is changed in the database; and an instruction executor for executing the instructions on the relational database and managing the target objects and the relationship in the database in accordance with the manipulation of the source object.

31. The manager as claimed in claim 30, wherein the instruction generator having an insert instruction generator of generating insert instructions to insert the multiple target objects into the corresponding target tables based on the mapping meta-data when a source object is inserted into the database.

32. The manager as claimed in claim 31, wherein the insert instruction generating function includes: a representation builder for building a database row representation for each target object; and a primary key information adder for adding the primary key value of the source object to the database row representation.

33. The manager as claimed in claim 31, wherein the instruction generator generates the insert instructions as SQL Insert statements.

34. The manager as claimed in claim 30, wherein the instruction generator having a read instruction generator for generating read instructions to read the multiple target objects and the relationship from the corresponding target tables and the foreign key values based on the mapping meta-data when a source object is read from the database.

35. The manager as claimed in claim 34, wherein the instruction generator generates the read instructions as SQL Select statements.

36. The manager as claimed in claim 30, wherein the instruction generator having a delete instruction generator for generating delete instructions to read the multiple target objects and the relationship from the corresponding target tables and the foreign key values based on the mapping meta-data when a source object is deleted from the database.

37. The manager as claimed in claim 36, wherein the instruction generator generates the delete instructions as SQL Delete statements.

38. (canceled)

39. The manager as claimed in claim 30, wherein the instruction generator generates the read instructions as SQL Insert statements, SQL Delete statements or SQL Update statements depending on the change to the source object.

40-46. (canceled)

47. A computer-readable storage medium storing instructions that when executed by a computer cause the computer to perform a method for managing object to relational one-to-many mapping for an object model mapped to a relational database, the method comprising a step of tracking changes to relationships and target objects with which the source object has one-to-many

relationships of privately owned type, wherein the tracking step comprises steps of: creating, at the start of a transaction, a source clone of a source object having a primary key value, a relationship clone of a relationship in the source object, and a target clone of target objects referenced by the relationship; determining, when the transaction is committed, changes to the source object and target objects by comparing the source clone to the current state of the source object, and comparing the target clone to the current state of the target objects using the primary key value of the source object added to the target objects, and updating the relationship between the source object and the target objects by: determining which objects have been added, removed or changed in the relationship by examining mapping meta-data including information of corresponding source table, one or more target tables and foreign keys of the target tables, generating an insert instruction for each target object of a source target that has been added, generating a delete instruction for each target object of a source target that has been removed, and generating an update instruction for each target object of a source target that has been changed.

48. The computer-readable storage medium as claimed in claim 47, wherein the steps of generating an insert, delete and update instructions comprise steps of: generating SQL Insert statements for objects that have been added; generating SQL Delete statements for objects that have been removed; and generating SQL Update statements for objects that have been changed.

49. The computer-readable storage medium as claimed in claim 47 further comprising a step of: storing the mapping meta-data external to the source object class and the target object classes.

50. The computer-readable storage medium as claimed in claim 49, wherein the storing step stores the mapping meta-data as XML files.

Previous Doc Next Doc Go to Doc#

L5: Entry 4 of 5 File: PGPB Feb 6, 2003

PGPUB-DOCUMENT-NUMBER: 20030028545
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20030028545 A1

TITLE: System and method for managing object to relational one-to-many mapping

PUBLICATION-DATE: February 6, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY
Wang, Yaoping	Ottawa		CA
Sutherland, James Bryce	Ottawa		CA

US-CL-CURRENT: 707/100

CLAIMS:

What is claimed is:

1. A method for managing object to relational one-to-many mapping for an object model mapped to a relational database, the method comprising steps of: obtaining, for a source object having a primary key value and being manipulated in a corresponding source table of the relational database, mapping meta-data including information of a corresponding target table for storing at least one target object with which the source object has a one-to-many relationship of privately owned type and information of a foreign key of the corresponding target table; generating an instruction to manipulate the at least one target object in the corresponding target table based on the mapping meta-data; and manipulating the at least one target object in the database by executing the instruction on the database.

2. The method as claimed in claim 1, wherein the instruction generating step generates an instruction to insert, read, delete or update the each target object in the database.

3. The method as claimed in claim 1, wherein each target object in the object model does not contain information regarding the one-to-many relationship.

4. The method as claimed in claim 1, wherein the generating step and the manipulating step are performed on each target object with which the source object has the one-to-many relationship of the privately owned type.

5. The method as claimed in claim 1, wherein the obtaining step is carried out when a system performing the method is initialized.

6. A method for managing object to relational one-to-many mapping for an object model mapped to a relational database, the method comprising steps of: obtaining, for a source object having a primary key value and being inserted in a corresponding source table of the relational database, mapping meta-data including information of a corresponding target table for storing at least one target object with which the source object has a one-to-many relationship of privately owned type and information of a foreign key of the corresponding target table;

generating an insert instruction to add a value of the foreign key based on the primary key value of the source object, and to insert the at least one target object into the corresponding target table based on the mapping meta-data; and inserting the at least one target object into the database by executing the insert instruction on the database.

7. The method as claimed in claim 6, wherein the insert instruction generating step and the target object inserting step comprise steps of: a) building a database row representation of each target object containing target object data; b) adding the primary key value of the source object to the database row representation; c) generating the insert instruction based on the database row representation and the target object data using the mapping meta-data; d) writing the target object in a row of the corresponding target table by executing the insert instruction on the database; and e) repeating steps a) to d) for each target object using the mapping meta-data.

8. The method as claimed in claim 7, wherein the adding step adds the primary key value of the source object as a foreign key value of the target object.

9. The method as claimed in claim 6, wherein the statement generating step generates the insert instructions as a Structured Query Language (SQL) Insert statement.

10. The method as claimed in claim 6 further comprising a step of: storing the mapping meta-data external to the source object class and the target object classes.

11. The method as claimed in claim 10, wherein the storing step stores the mapping meta-data as XML files.

12. The method as claimed in claim 6 further comprising steps of: reading the source object from the object model; and inserting the source object into the corresponding source table in the database.

13. A method for managing object to relational one-to-many mapping for an object model mapped to a relational database, the method comprising steps of: obtaining, when a source object having a primary key value is being read from a source table in the relational database, mapping meta-data including information of one or more corresponding target tables and information of one or more foreign keys of the corresponding target tables; generating a select instruction to select from the target tables target objects with which the source object has one-to-many relationships of privately owned type, based on the mapping meta-data and the primary key value of the source object; and reading the target objects and relationships relating to the source object from the database by executing the select instruction on the database.

14. The method as claimed in claim 13, wherein the reading step comprises steps of: querying for rows in the target tables that have a foreign key value matching the primary key value of the source object by executing the select instruction on the database; translating the queried rows into target objects based on the mapping meta-data; adding the target objects to a collection that represents a value of relationship of the source object, the value of relationship referencing to the target objects; and setting the value of the relationship into the source object.

15. The method as claimed in claim 13, wherein the generating step comprises a step of generating a select instruction to check foreign key fields in the target tables.

16. The method as claimed in claim 13, wherein the statement generating step generates the read instructions as SQL Select statements.

17. The method as claimed in claim 16 further comprising a step of: storing the mapping meta-data external to the source object class and the target object classes.

18. The method as claimed in claim 17, wherein the storing step stores the mapping meta-data as XML files.

19. A method for managing object to relational one-to-many mapping for an object model mapped to a relational database, the method comprising steps of: obtaining, when a source object having a primary key value is being deleted from a source table in the relational database; mapping meta-data that defines one or more corresponding target tables storing target objects with which the source object has one-to-many relationships of privately owned type and foreign key information; generating a delete instruction to delete the target objects from the target tables based on the mapping meta-data; and deleting the target objects by executing the delete instruction on the database.

20. The method as claimed in claim 19 further comprising a step of determining whether the target objects are stored in a single target table, and wherein, when the target objects are determined to be stored in a single target table, the delete instruction generating step comprises a step of generating a single delete instruction that deletes all rows from the target table that have a foreign key matching the primary key of the source object.

21. The method as claimed in claim 19 further comprising a step of determining whether the target objects are stored in multiple target tables; and wherein, when the target objects are determined to be stored in multiple target tables, the delete instruction generating step comprises steps of: extracting a primary key value for each target object; generating a delete instruction to delete each row with the primary key value; and repeating the extracting step and the generating step.

22. The method as claimed in claim 19, wherein the statement generating step generates the delete instructions as SQL Delete statements.

23. The method as claimed in claim 19 further comprising a step of: storing the mapping meta-data external to the source object class and the target object classes.

24. The method as claimed in claim 23, wherein the storing step stores the mapping meta-data as XML files.

25. A method for managing object to relational one-to-many mapping for an object model mapped to a relational database, the method comprising a step of tracking changes to relationships and target objects with which the source object has one-to-many relationships of privately owned type, wherein the tracking step comprises steps of: creating, at the start of a transaction, a source clone of a source object having a primary key value, a relationship clone of a relationship in the source object, and a target clone of target objects referenced by the relationship; and determining, when the transaction is committed, changes to the source object and target objects by comparing the source clone to the current state of the source object, and comparing the target clone to the current state of the target objects using the primary key value of the source object added to the target objects.

26. The method as claimed in claim 25 further comprising a step of updating relationship of the source object referencing target objects to the database wherein the updating step comprises steps of: determining which objects have been added, removed or changed in the relationship using mapping meta-data including information of corresponding source table, one or more target tables and foreign keys of the target tables; generating an insert instruction for each target object of a source target that has been added; generating a delete instruction for each target object of a source target that has been removed; and generating an update instruction for each target object of a source target that has been changed.

27. The method as claimed in claim 25, wherein the steps of generating an insert, delete and update instructions comprise steps of: generating SQL Insert statements for objects that have been added; generating SQL Delete statements for objects that have been removed; and generating SQL Update statements for objects that have been changed.

28. The method as claimed in claim 25 further comprising a step of: storing the mapping meta-data external to the source object class and the target object classes.

29. The method as claimed in claim 28, wherein the storing step stores the mapping meta-data as XML files.

30. A one-to-many mapping manager for managing object to relational one-to-many mapping for an object model mapped to a relational database, the system comprising: a meta-data receiver for obtaining, for a source data having a primary key value and being manipulated, mapping meta-data including information of one or more corresponding target tables for storing multiple target objects with which the source object has one-to-many relationships of privately owned type and information of one or more foreign keys of the corresponding target tables; an instruction generator for generating instructions to manage the multiple target objects and the relationship based on the mapping meta-data; and an instruction executor for executing the instructions on the relational database and managing the target objects and the relationship in the database in accordance with the manipulation of the source object.

31. The manager as claimed in claim 30, wherein the instruction generator having an insert instruction generator of generating insert instructions to insert the multiple target objects into the corresponding target tables based on the mapping meta-data when a source object is inserted into the database.

32. The manager as claimed in claim 31, wherein the insert instruction generating function includes: a representation builder for building a database row representation for each target object; and a primary key information adder for adding the primary key value of the source object to the database row representation.

33. The manager as claimed in claim 31, wherein the instruction generator generates the insert instructions as SQL Insert statements.

34. The manager as claimed in claim 30, wherein the instruction generator having a read instruction generator for generating read instructions to read the multiple target objects and the relationship from the corresponding target tables and the foreign key values based on the mapping meta-data when a source object is read from the database.

35. The manager as claimed in claim 34, wherein the instruction generator generates the read instructions as SQL Select statements.

36. The manager as claimed in claim 30, wherein the instruction generator having a delete instruction generator for generating delete instructions to read the multiple target objects and the relationship from the corresponding target tables and the foreign key values based on the mapping meta-data when a source object is deleted from the database.

37. The manager as claimed in claim 36, wherein the instruction generator generates the delete instructions as SQL Delete statements.

38. The manager as claimed in claim 30, wherein the instruction generator having an update instruction generator for generating update instructions to track changes to the multiple target objects and the relationship, and to update the database when a source object is changed in the database.

39. The manager as claimed in claim 38, wherein the instruction generator generates the read instructions as SQL Insert statements, SQL Delete statements or SQL Update statements depending on the change to the source object.

40. A mapping system for managing an object model to a relational database, the object model containing object classes having one or more source objects and target objects having one-to-

many relationships of privately owned type with the source objects, the relational database having tables including one or more source tables for storing the source objects and one or more target tables for storing the target objects, the source tables having primary keys and the target tables having foreign keys, the system comprising: a mapping tool for assisting in mapping object classes to tables and mapping relationships to foreign keys; a meta-data storage for storing mapping meta-data defining information of how object classes map to tables and information of how relationships map to foreign keys, the mapping meta-data including information of one or more corresponding target tables for storing the multiple target objects and information of one or more foreign keys of the corresponding target tables; and a runtime mapping library for accessing the mapping meta-data in the meta-data storage and managing object data in the database, the runtime mapping library having a one-to-may mapping manager for managing one-to-many relationships and target objects with which a source objects have one-to-many relationships in accordance with the manipulation of the source object.

41. The mapping system as claimed in claim 40, wherein the one-to-many mapping manager comprises: an instruction generator for generating instructions to manage the multiple target objects and the relationship based on the mapping meta-data; and an instruction executor for executing the instructions on the database and managing the target objects and the relationship in the database in accordance with the manipulation of the source object.

42. The manager as claimed in claim 40, wherein the meta-data storage is provided external to the source object class and the target object classes.

43. The manager as claimed in claim 40, wherein the meta-data storage is an XML file system for storing the mapping meta-data in XML files.

44. Computer readable media storing the instructions or statements for use in the execution in a computer of a method for managing object to relational one-to-many mapping for an object model mapped to a relational database, the method comprising steps of: obtaining, for a source object having a primary key value and being manipulated in a corresponding source table of the relational database, mapping meta-data including information of a corresponding target table for storing at least one target object with which the source object has a one-to-many relationship of privately owned type and information of a foreign key of the corresponding target table; generating an instruction to manipulate the at least one target object in the corresponding target table based on the mapping meta-data; and manipulating the at least one target object in the database by executing the instruction on the database.

45. Electronic signals for use in the execution in a computer of a method for managing object to relational one-to-many mapping for an object model mapped to a relational database, the method comprising steps of: obtaining, for a source object having a primary key value and being manipulated in a corresponding source table of the relational database, mapping meta-data including information of a corresponding target table for storing at least one target object with which the source object has a one-to-many relationship of privately owned type and information of a foreign key of the corresponding target table; generating an instruction to manipulate the at least one target object in the corresponding target table based on the mapping meta-data; and manipulating the at least one target object in the database by executing the instruction on the database.

46. A computer program product for use in the execution in a computer of a method for managing object to relational one-to-many mapping for an object model mapped to a relational database, the computer program product comprising: a module for obtaining, for a source object having a primary key value and being manipulated in a corresponding source table of the relational database, mapping meta-data including information of a corresponding target table for storing at least one target object with which the source object has a one-to-many relationship of privately owned and information of a foreign key of the corresponding target table; a module for generating an instruction to manipulate the at least one target object in the corresponding target table based on the mapping meta-data; and a module for manipulating the at least one target object in the database by executing the instruction on the database.